Concepts of CSS

What is CSS?

Cascading Style Sheets (CSS) is a stylesheet language used for describing the presentation and styling of documents written in markup languages like HTML. CSS is a collection of formatting rules that control the appearance of content in a web page. It separates content from presentation, making web development more efficient and maintainable.

Key Features and Benefits of CSS

Primary Purpose:

- Enable separation of presentation and content
- Improve content accessibility
- Provide flexibility and control in presentation
- Reduce complexity and repetition in structural content

Advantages of CSS:

- Simplifies web design and maintenance
- Enhances website performance and user experience
- Supports responsive and adaptive designs for all devices
- Promotes code reusability and consistency
- Results in cleaner HTML code and faster loading times

Understanding "Cascading"

The term **"cascading"** refers to how styles are applied to elements based on priority rules[66]. When multiple CSS rules target the same element, the browser follows a cascading order to determine which style to apply.

CSS Priority Scheme (Highest to Lowest):

Priority	CSS Source Type	Description
1	Importance	The "!important" annotation overwrites previous priority types

2	Inline	Style applied via HTML "style" attribute
3	Media Type	Property definition applies to all media types unless media-specific CSS is defined
4	User defined	User-defined CSS for accessibility features
5	Selector specificity	Specific contextual selectors override generic definitions
6	Rule order	Last rule declaration has higher priority
7	Parent inheritance	Properties inherited from parent elements
8	CSS in HTML document	CSS rules override default browser values
9	Browser default	Lowest priority: browser default values

Creating Style Sheets

Types of CSS Implementation

1. Inline CSS:

- CSS applied directly within HTML elements using the `style` attribute
- Highest priority in cascading hierarchy
- Quick to apply but not recommended for multiple elements

```
<p style="color: blue; font-size: 16px;">This is inline CSS</p>
```

2. Internal CSS:

- CSS written within `<style>` tags in the `<head>` section
- Useful for single-page styling
- Applies to entire document

```
<head>
    <style>
        p {
            color: blue;
            font-size: 16px;
        }
    </style>
```

```
</head>
```

3. External CSS:

- Most efficient method for larger sites
- CSS written in separate `.css` file
- Linked to HTML using `<link>` tag
- Promotes clean, manageable code

```
<head>
    <link rel="stylesheet" type="text/css" href="styles.css">
</head>
```

CSS Syntax Structure

Basic CSS Syntax:

```
selector {
    property: value;
    property: value;
}
```

Components:

- **Selector**: Points to HTML element to style
- **Property**: Aspect of element to change
- **Value**: Setting to apply to the property

CSS Properties and Styling

Background Styling Properties

Background Color:

```
div {
    background-color: #b0c4de;
    /* or */
```

```
    background-color: rgb(255, 0, 0);
    /* or */
    background-color: red;
}
```

Background Image:

```
body {
    background-image: url('image.gif');
}
```

Background Repeat:

```
.element {
    background-repeat: repeat-x;    /* Horizontal repeat */
    background-repeat: repeat-y;    /* Vertical repeat */
    background-repeat: no-repeat;   /* No repeat */
}
```

Background Position:

```
.element {
    background-position: right top;
    background-position: center center;
    background-position: 50px 100px;
}
```

Background Attachment:

```
.element {
    background-attachment: scroll;  /* Scrolls with content */
    background-attachment: fixed;   /* Fixed position */
}
```

Background Shorthand Property:

```
.element {
    background: #ffffff url('img_tree.png') no-repeat right top;
```

```
}
```

Text Formatting Properties

Text Color:

```
p {
    color: red;           /* Color name */
    color: #ff0000;       /* HEX value */
    color: rgb(255,0,0);  /* RGB value */
}
```

Text Alignment:

```
h1 { text-align: center; }
p { text-align: left; }
div { text-align: right; }
.justify { text-align: justify; }
```

Text Decoration:

```
.underline { text-decoration: underline; }
.overline { text-decoration: overline; }
.strikethrough { text-decoration: line-through; }
.none { text-decoration: none; }
```

Text Transform:

```
.uppercase { text-transform: uppercase; }
.lowercase { text-transform: lowercase; }
.capitalize { text-transform: capitalize; }
```

Controlling Fonts

Font Family:

```
p {
    font-family: "Times New Roman", Times, serif;
    font-family: Arial, Helvetica, sans-serif;
```

```
}
```

Font Size:

```
h1 { font-size: 2rem; }
p { font-size: 16px; }
span { font-size: 1.2em; }
```

Font Weight:

```
.normal { font-weight: normal; }
.bold { font-weight: bold; }
.bolder { font-weight: bolder; }
.light { font-weight: 300; }
```

Font Style:

```
.normal { font-style: normal; }
.italic { font-style: italic; }
.oblique { font-style: oblique; }
```

Working with Block Elements and Objects

Understanding Block vs Inline Elements

Block Elements:

- Always start on a new line
- Take up full width available
- Have margins automatically added
- Examples: `<div>`, `<p>`, `<h1>-<h6>`, `<section>`, `<header>`, `<footer>`

```
div {
    display: block;
    width: 100%;
    margin: 10px 0;
}
```

Common Block Elements:

- `<div>` – Generic container
- `<p>` – Paragraph
- `<h1>` to `<h6>` – Headings
- `<ul>` and `<ol>` – Lists
- `<table>` – Tables
- `<section>` – Thematic content groups
- `<header>` and `<footer>` – Page headers/footers

Display Property

Display Values:

```
.block { display: block; }
.inline { display: inline; }
.inline-block { display: inline-block; }
.none { display: none; }
.flex { display: flex; }
.grid { display: grid; }
```

Key Differences:

- `display: block` – Extends to 100% of available space
- `display: table` – Only as wide as its contents
- `display: inline-block` – Combines features of inline and block elements[71]

Working with Lists and Tables

Styling Lists

List Style Properties:

```
ul {
    list-style-type: disc;     /* bullet, circle, square, none */
    list-style-position: inside; /* inside, outside */
    list-style-image: url('bullet.png');
```

```
}

/* Shorthand */
ul {
    list-style: square inside url('bullet.png');
}
```

Custom List Styling:

```
.custom-list {
    list-style-type: none;
    padding-left: 0;
}

.custom-list li {
    background: url('custom-bullet.png') no-repeat left center;
    padding-left: 20px;
    margin: 5px 0;
}
```

Styling Tables

Table Structure Styling:

```
table {
    border-collapse: collapse;
    width: 100%;
    margin: 20px 0;
}

th, td {
    border: 1px solid #ddd;
    padding: 8px 12px;
    text-align: left;
}

th {
    background-color: #f2f2f2;
    font-weight: bold;
```

```
}

tr:nth-child(even) {
    background-color: #f9f9f9;
}

tr:hover {
    background-color: #f5f5f5;
}
```

CSS ID and Class Selectors

Class Selectors

Syntax and Usage:

```
.class-name {
    property: value;
}
```

HTML Implementation:

```
<div class="highlight">This div has a class</div>
<p class="text-large text-center">Multiple classes</p>
```

CSS Styling:

```
.highlight {
    background-color: yellow;
    padding: 10px;
}

.text-large {
    font-size: 18px;
}

.text-center {
    text-align: center;
```

```
}
```

ID Selectors

Syntax and Usage:

```
#id-name {
    property: value;
}
```

HTML Implementation:

```
<div id="header">Unique header element</div>
<p id="main-content">Main content paragraph</p>
```

CSS Styling:

```
#header {
    background-color: #333;
    color: white;
    padding: 20px;
}

#main-content {
    font-size: 16px;
    line-height: 1.6;
}
```

Key Differences Between Class and ID

Feature	Class Selector	ID Selector
Syntax	`.class-name`	`#id-name`
HTML Attribute	`class="example"`	`id="example"`
Reusability	Multiple elements	Single element only
Specificity	Lower priority	Higher priority
Use Case	Reusable styles	Unique elements

Combining Selectors:

```
/* Element with specific class */
p.highlight { color: red; }

/* Element with specific ID */
div#container { width: 100%; }

/* Multiple classes */
.btn.primary { background-color: blue; }
```

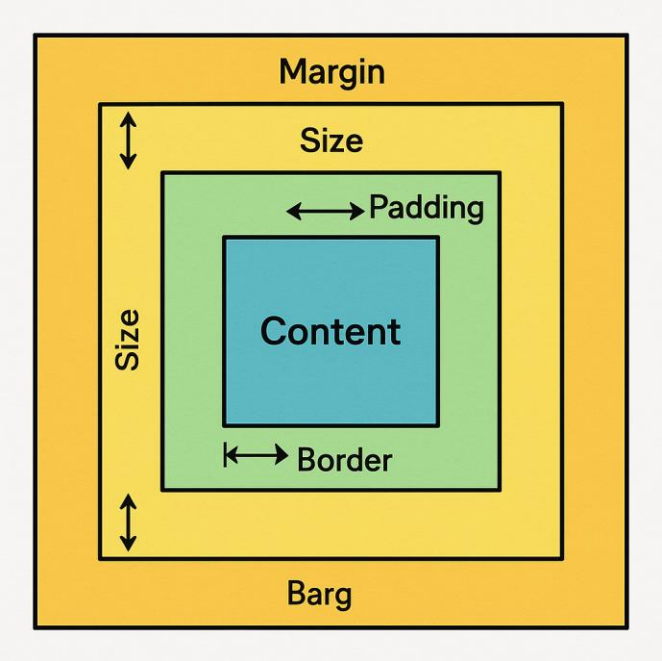
Box Model

Introduction to Box Model

The **CSS Box Model** describes how every visible element on a webpage is represented as a rectangular box. Each box consists of four areas:

1. **Content Area**: Contains text, images, and other content
2. **Padding**: Space around content, inside the border
3. **Border**: Surrounds the padding and content
4. **Margin**: Outermost area, space outside the border

Box Model Visualization



Border Properties

Border Syntax:

```
border: width style color;
```

Individual Border Properties:

```
.element {
    border-width: 2px;
    border-style: solid;    /* solid, dashed, dotted, double */
    border-color: red;

    /* Individual sides */
    border-top: 1px solid black;
    border-right: 2px dashed blue;
    border-bottom: 3px dotted green;
    border-left: 4px double red;
}
```

Border Style Values:

- `solid` – Solid line
- `dashed` – Dashed line
- `dotted` – Dotted line
- `double` – Double line
- `groove` – 3D grooved border
- `ridge` – 3D ridged border
- `inset` – 3D inset border
- `outset` – 3D outset border

Padding Properties

Padding controls the space inside the border[81]:

```
/* All sides */
.element { padding: 20px; }
```

```
/* Vertical | Horizontal */
.element { padding: 10px 20px; }

/* Top | Horizontal | Bottom */
.element { padding: 10px 20px 15px; }

/* Top | Right | Bottom | Left */
.element { padding: 10px 20px 15px 25px; }

/* Individual sides */
.element {
    padding-top: 10px;
    padding-right: 20px;
    padding-bottom: 15px;
    padding-left: 25px;
}
```

Margin Properties

Margin controls the space outside the border:

```
/* All sides */
.element { margin: 20px; }

/* Vertical | Horizontal */
.element { margin: 10px 20px; }

/* Top | Horizontal | Bottom */
.element { margin: 10px 20px 15px; }

/* Top | Right | Bottom | Left */
.element { margin: 10px 20px 15px 25px; }

/* Individual sides */
.element {
    margin-top: 10px;
    margin-right: 20px;
    margin-bottom: 15px;
```

```
    margin-left: 25px;
}

/* Centering elements */
.center { margin: 0 auto; }
```

Box-Sizing Property

Content-Box (Default):

```
.content-box {
    box-sizing: content-box;
    width: 200px;
    padding: 20px;
    border: 5px solid black;
    /* Total width = 200px + 40px + 10px = 250px */
}
```

Border-Box:

```
.border-box {
    box-sizing: border-box;
    width: 200px;
    padding: 20px;
    border: 5px solid black;
    /* Total width = 200px (includes padding and border) */
}
```

CSS Color, Grouping, and Dimensions

Color Properties

Color Value Types :

```
.element {
    /* Named colors */
    color: red;
    background-color: blue;
```

```
    /* Hexadecimal */
    color: #ff0000;
    background-color: #0000ff;

    /* RGB */
    color: rgb(255, 0, 0);
    background-color: rgb(0, 0, 255);

    /* RGBA (with transparency) */
    color: rgba(255, 0, 0, 0.5);
    background-color: rgba(0, 0, 255, 0.8);

    /* HSL */
    color: hsl(0, 100%, 50%);
    background-color: hsl(240, 100%, 50%);
}
```

Grouping Selectors

Multiple Selectors:

```
/* Group selectors with comma */
h1, h2, h3 {
    color: blue;
    font-family: Arial, sans-serif;
}

/* Descendant selectors */
nav ul li {
    list-style: none;
    display: inline;
}

/* Child selectors */
.container > .item {
    margin: 10px;
}

/* Adjacent sibling */
```

```
h1 + p {
    font-size: 18px;
}
```

Dimension Properties

Width and Height:

```
.element {
    width: 300px;
    height: 200px;
    max-width: 100%;
    min-height: 150px;

    /* Percentage values */
    width: 50%;
    height: 80vh;

    /* Auto values */
    width: auto;
    height: auto;
}
```

Display, Positioning, and Floating

Display Properties

Common Display Values:

```
.block { display: block; }
.inline { display: inline; }
.inline-block { display: inline-block; }
.flex { display: flex; }
.grid { display: grid; }
.table { display: table; }
.none { display: none; }
```

Positioning

Position Property Values:

1. Static (Default):

```
.static {
    position: static;
    /* Not affected by top, bottom, left, right */
}
```

2. Relative:

```
.relative {
    position: relative;
    top: 20px;
    left: 30px;
    /* Positioned relative to normal position */
}
```

3. Absolute:

```
.absolute {
    position: absolute;
    top: 50px;
    right: 100px;
    /* Positioned relative to nearest positioned ancestor */
}
```

4. Fixed:

```
.fixed {
    position: fixed;
    bottom: 0;
    right: 0;
    /* Positioned relative to viewport */
}
```

5. Sticky:

```
.sticky {
    position: sticky;
    top: 20px;
    /* Switches between relative and fixed based on scroll */
}
```

Floating

Float Property:

```
.float-left {
    float: left;
    width: 200px;
    margin: 10px;
}

.float-right {
    float: right;
    width: 200px;
    margin: 10px;
}

.clear-both {
    clear: both;
}
```

Float vs Text-Align:

- `text-align` applies to content within an element
- `float` applies to the element itself
- `float` removes element from normal document flow

Alignment Properties

Text Alignment

Horizontal Text Alignment:

```
.left { text-align: left; }
.center { text-align: center; }
.right { text-align: right; }
.justify { text-align: justify; }
```

Element Alignment

Centering Elements:

```
/* Horizontal centering */
.center-block {
    margin: 0 auto;
    width: 300px;
}

/* Flexbox centering */
.flex-center {
    display: flex;
    justify-content: center;
    align-items: center;
}

/* Absolute centering */
.absolute-center {
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);
}
```

Pseudo-Classes

Common Pseudo-Classes

Link Pseudo-Classes[107]:

```
a:link { color: blue; }        /* Unvisited links */
a:visited { color: purple; }   /* Visited links */
a:hover { color: red; }        /* Mouse over */
```

```
a:active { color: orange; }    /* Active/clicked */
```

Interactive Pseudo-Classes:

```
button:hover {
    background-color: lightblue;
    cursor: pointer;
}

input:focus {
    border: 2px solid blue;
    outline: none;
}

button:active {
    transform: scale(0.98);
}
```

Structural Pseudo-Classes:

```
p:first-child { font-weight: bold; }
p:last-child { margin-bottom: 0; }
tr:nth-child(even) { background-color: #f2f2f2; }
tr:nth-child(odd) { background-color: white; }

/* Advanced nth-child */
li:nth-child(3n+1) { color: red; }    /* 1st, 4th, 7th... */
p:not(.special) { color: black; }      /* All p except .special */
```

Form Pseudo-Classes

```
input:valid {
    border: 2px solid green;
}

input:invalid {
    border: 2px solid red;
}
```

```
input:required {
    background-color: lightyellow;
}

input:disabled {
    opacity: 0.5;
    cursor: not-allowed;
}
```

Navigation Bar

Basic Navigation Structure

HTML Structure:

```
<nav class="navbar">
    <ul class="nav-list">
        <li class="nav-item"><a href="#home" class="nav-link">Home</a></li>
        <li class="nav-item"><a href="#about" class="nav-link">About</a></li>
        <li class="nav-item"><a href="#services" class="nav-link">Services</a></li>
        <li class="nav-item"><a href="#contact" class="nav-link">Contact</a></li>
    </ul>
</nav>
```

Horizontal Navigation Bar

```
.navbar {
    background-color: #333;
    padding: 0;
    margin: 0;
}

.nav-list {
    list-style-type: none;
    margin: 0;
    padding: 0;
    display: flex;
}
```

```
.nav-item {
    border-right: 1px solid #555;
}

.nav-link {
    display: block;
    color: white;
    text-decoration: none;
    padding: 15px 20px;
    transition: background-color 0.3s ease;
}

.nav-link:hover {
    background-color: #555;
}

.nav-link:active,
.nav-link.active {
    background-color: #04AA6D;
}
```

Vertical Navigation Bar

```
.vertical-nav {
    width: 200px;
    background-color: #f1f1f1;
    height: 100vh;
    position: fixed;
    top: 0;
    left: 0;
}

.vertical-nav .nav-list {
    flex-direction: column;
}

.vertical-nav .nav-item {
    border-bottom: 1px solid #ccc;
    border-right: none;
```

```
}
```

Responsive Navigation

```
@media screen and (max-width: 768px) {
    .nav-list {
        flex-direction: column;
    }

    .nav-item {
        border-right: none;
        border-bottom: 1px solid #555;
    }
}
```

Image Sprites

What are Image Sprites?

Image sprites are a technique that combines multiple small images into one larger image file. This reduces the number of HTTP requests and improves website performance.

Benefits of Image Sprites

- **Reduced HTTP Requests**: Multiple images loaded with single request
- **Faster Loading**: Fewer server requests mean faster page load times
- **Bandwidth Efficiency**: Less overall data transfer
- **Caching Benefits**: Single image file cached by browser

Creating Image Sprites

Step 1: Prepare Sprite Image:
Create a single image file containing all your icons arranged in a grid pattern.

Step 2: Base Sprite Class:

```
.sprite {
    background-image: url('images/sprite.png');
```

```
    background-repeat: no-repeat;
    display: inline-block;
}
```

Step 3: Individual Icon Classes:

```
.icon-home {
    width: 50px;
    height: 50px;
    background-position: 0 0;
}

.icon-about {
    width: 50px;
    height: 50px;
    background-position: -50px 0;
}

.icon-services {
    width: 50px;
    height: 50px;
    background-position: -100px 0;
}

.icon-contact {
    width: 50px;
    height: 50px;
    background-position: -150px 0;
}
```

HTML Implementation

```
<div class="sprite icon-home"></div>
<div class="sprite icon-about"></div>
<div class="sprite icon-services"></div>
<div class="sprite icon-contact"></div>
```

Advanced Sprite Techniques

Hover Effects with Sprites:

```
.sprite-button {
    width: 100px;
    height: 40px;
    background-image: url('button-sprite.png');
    background-position: 0 0;
    border: none;
    cursor: pointer;
}

.sprite-button:hover {
    background-position: 0 -40px;
}

.sprite-button:active {
    background-position: 0 -80px;
}
```

Calculating Background Position

Formula for Position Calculation:

- **Horizontal Position**: `-(icon_index_x * icon_width)px`
- **Vertical Position**: `-(icon_index_y * icon_height)px`

Example:

If each icon is 50px × 50px and you want the icon in position (2,1):

```
.icon {
    background-position: -100px -50px; /* -(2*50)px -(1*50)px */
}
```

Complete CSS Example

```
/* CSS Reset */
* {
    margin: 0;
    padding: 0;
```

```
    box-sizing: border-box;
}

/* Body Styling */
body {
    font-family: Arial, sans-serif;
    line-height: 1.6;
    color: #333;
    background-color: #f4f4f4;
}

/* Container */
.container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 20px;
}

/* Header */
header {
    background-color: #333;
    color: white;
    padding: 1rem 0;
    position: sticky;
    top: 0;
    z-index: 100;
}

.header-content {
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.logo {
    font-size: 1.5rem;
    font-weight: bold;
}

/* Navigation */
```

```
.nav-list {
    display: flex;
    list-style: none;
    gap: 2rem;
}

.nav-link {
    color: white;
    text-decoration: none;
    padding: 0.5rem 1rem;
    border-radius: 4px;
    transition: background-color 0.3s ease;
}

.nav-link:hover {
    background-color: #555;
}

/* Main Content */
main {
    background-color: white;
    margin: 2rem 0;
    padding: 2rem;
    border-radius: 8px;
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);
}

/* Cards */
.card-container {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
    gap: 2rem;
    margin: 2rem 0;
}

.card {
    border: 1px solid #ddd;
    border-radius: 8px;
    overflow: hidden;
    transition: transform 0.3s ease, box-shadow 0.3s ease;
```

```
}

.card:hover {
    transform: translateY(-5px);
    box-shadow: 0 4px 15px rgba(0,0,0,0.1);
}

.card-header {
    background-color: #007bff;
    color: white;
    padding: 1rem;
}

.card-body {
    padding: 1.5rem;
}

.card-title {
    margin-bottom: 1rem;
    color: #333;
}

/* Buttons */
.btn {
    display: inline-block;
    padding: 0.75rem 1.5rem;
    background-color: #007bff;
    color: white;
    text-decoration: none;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}

.btn:hover {
    background-color: #0056b3;
}

.btn-secondary {
```

```
    background-color: #6c757d;
}

.btn-secondary:hover {
    background-color: #545b62;
}

/* Forms */
.form-group {
    margin-bottom: 1rem;
}

.form-label {
    display: block;
    margin-bottom: 0.5rem;
    font-weight: bold;
}

.form-control {
    width: 100%;
    padding: 0.75rem;
    border: 1px solid #ddd;
    border-radius: 4px;
    font-size: 1rem;
}

.form-control:focus {
    outline: none;
    border-color: #007bff;
    box-shadow: 0 0 0 2px rgba(0, 123, 255, 0.25);
}

/* Tables */
.table {
    width: 100%;
    border-collapse: collapse;
    margin: 1rem 0;
}

.table th,
```

```
.table td {
    padding: 0.75rem;
    text-align: left;
    border-bottom: 1px solid #ddd;
}

.table th {
    background-color: #f8f9fa;
    font-weight: bold;
}

.table tbody tr:hover {
    background-color: #f5f5f5;
}

/* Footer */
footer {
    background-color: #333;
    color: white;
    text-align: center;
    padding: 2rem 0;
    margin-top: 2rem;
}

/* Responsive Design */
@media (max-width: 768px) {
    .header-content {
        flex-direction: column;
        gap: 1rem;
    }

    .nav-list {
        flex-direction: column;
        width: 100%;
        text-align: center;
    }

    .container {
        padding: 1rem;
    }
```

```
    .card-container {
        grid-template-columns: 1fr;
    }
}
```

This comprehensive guide covers all essential CSS concepts from basic styling to advanced techniques like image sprites and responsive navigation bars. Each section includes practical examples and real-world applications to help you master CSS development.